EleNA Design Document

I. Introduction

Navigation systems that we use in our day-to-day life helps us reach our required location though we are unaware of the routes to be taken, thus helping us avoid the criteria of being lost. The existing navigation works by finding out the shortest possible path between two given locations. Though there are certain developments in current applications of google maps, apple maps in showing multiple routes to reach a particular destination. There are certain functionalities when added could improve the usability of the application. As the steeps and lows of the path may be hard to traverse, for certain users, traveling comfortably on a plain path may be more important than traveling on the shortest way with elevation. Information regarding the elevation and extra distance (% value by which the selected route is more than the shortest distance) may be useful for users who engage in any highly strenuous training exercises that include climbing up or down hills. So, we have come up with an application – ELeNA which stands for Elevation based Navigation System.

II. Objective

The designed web-application which is user interactive Elevation based Navigation System is intended to deliver the functionalities such as enabling the user to select the elevation of the path as maximum or minimum and to give the % distance of the shortest path that they would like to travel.

i. Input Data Control: The user can give the inputs of the required start and end location.
ii. Elevation Control: We can choose the elevation to be maximum or minimum as per the requirements.
iii. x% of shortest path: we specify the percentage of shortest path that we should travel.

III. Requirement Specifications

Functional Requirements:

Functional requirements of the application basically define what it is expected to do, specific features and functions. It emphasizes on how the system is expected to work and the business rules it must follow. While designing the system, the following functional requirements are kept in thought.

i. The user interface should be interactive.
ii. User should be able to give a valid start and end locations and submit the route request.
iii. The elevation gain could be maximum or minimum as per the user requirements (a toggle button is provided for the same.
iv. A slider which denotes x% of the required path to be taken is also provided in the Front-end.
v. An algorithm is implemented such that it computes the route based on the given inputs and the respective route statistics (elevation gain and total distance) are displayed.
vi. A map that shows the route between the start location and end location as per the required inputs.
vii. The map is magnifiable and movable.

Non-Functional Requirements:

The non-functional requirements help us understand the quality of the system. The non-functional requirements that we have planned to include involves:

i. Readability.
ii. Usability.
iii. Debuggability.
iv. Version control.
v. Modularity.
vi. Performance.
vii. Reliability, maintainability.

IV. Methodology

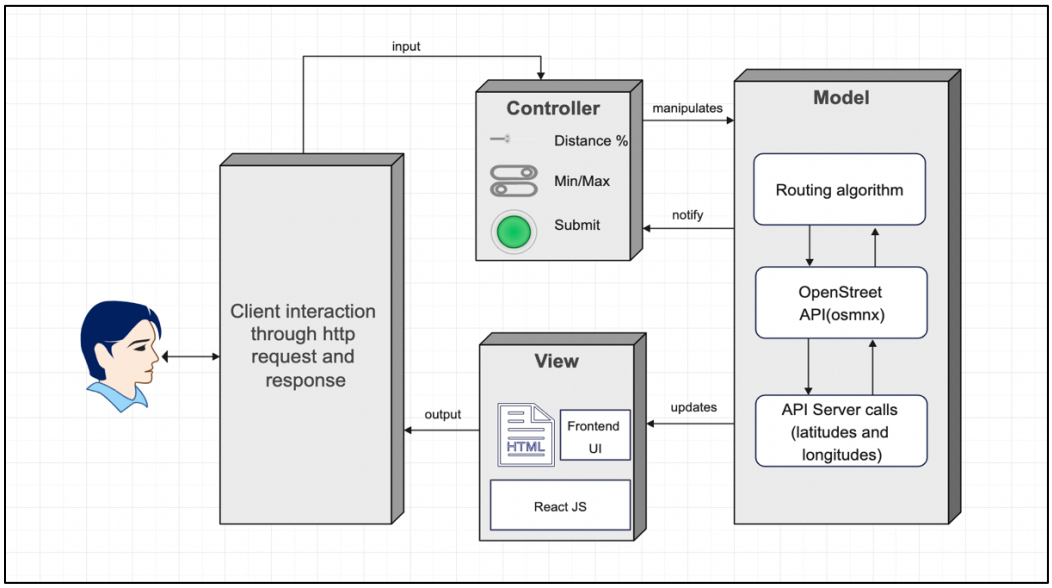


Fig 1. Architecture of ELeNA

In ELeNA the application is made much usable by inculcating the elevation factor in the existing navigation system. It is developed using multiple software tools and languages such as IntelliJ, Visual Studios, Reactjs, python, HTML, CSS and then deployed in MacOS with a model, view, and a controller architecture as it helps in breaking up the code into Front-end and Back-end to manage them easily and further make the changes separately. It is one of the most frequently deployed frameworks to deliver the projects involving the web development. Updation of the application is easier with MVC architecture as it allows multiple collaborators to work simultaneously. As each level and components of the code is properly deployed it enables the developer to debug it easily. IntelliJ IDEA Platform is a platform for building smart, language aware integrated development environment (IDE) with a comprehensive set of components used for developing software applications. Among all the modern front-end frameworks available Reactjs is most preferred one since it is very a simple and lightweight library that only deals with the view layer. React is a popular JavaScript library developed by Facebook for building web application user interfaces. This component-based structure is helpful in building tremendously large web applications with consistent look and feel. The major advantage of this software is code reusability which makes it easy for a developer to maintain and grow the code. Visual Studio is an IDE that enables to write code accurately and efficiently without losing the current file context.

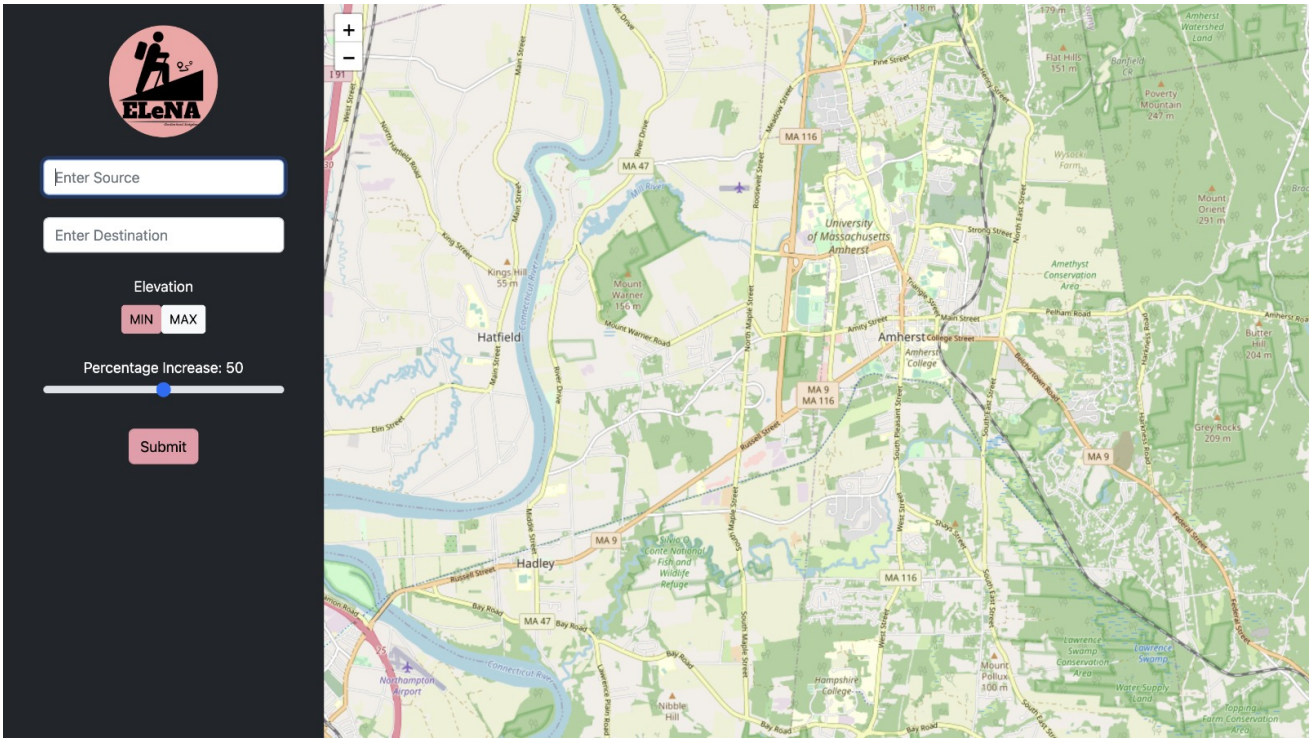
The Visual Studio Code editor supports React.js, code navigation out of the box.HTML (Hypertext Markup Language) – is used to structure web pages and CSS (Cascading Style Sheets) is used for proper presentation of the web page.HTML defines the presence of elements in the web page, such as various hyperlinks and files of media. It is easier to understand, and all the browsers supports HTML, while CSS is used for adjusting the fonts, colors, and various presentation artifacts.

V. Implementation Details

The entire web application works with the inputs of the user. It has three segments, view, controller, and model.

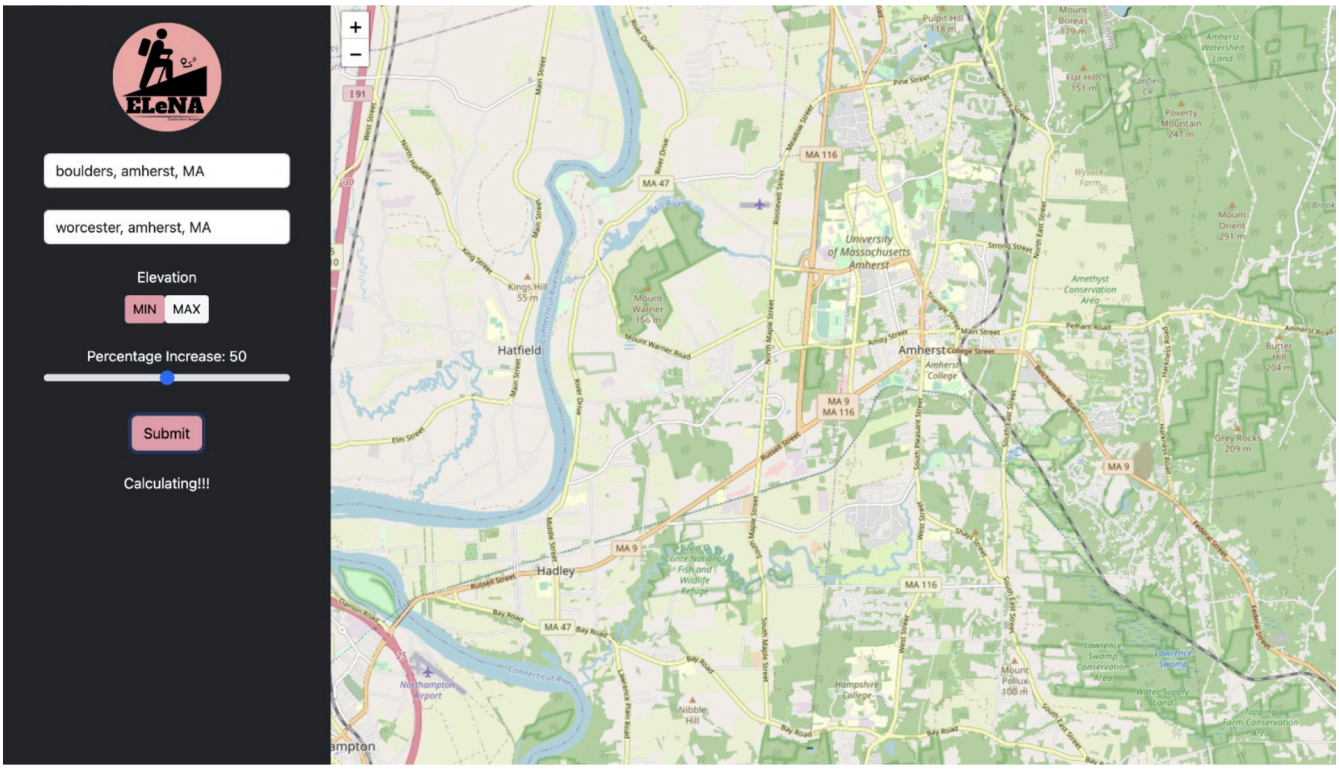
The three stages of working of the web application.

Stage 1:



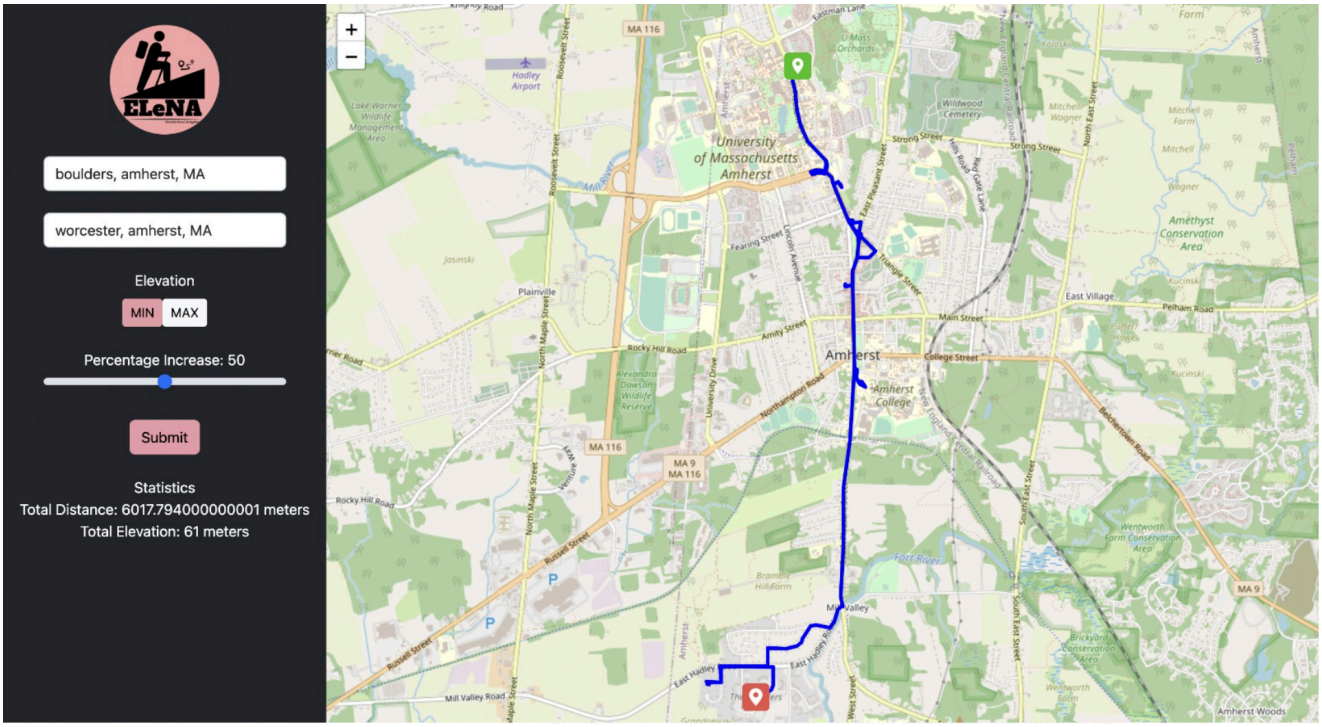
This is the initial view page that a user views as soon as the application is opened. The user is expected to provide his start, end locations and the other specifications of elevation, percentage increase of the actual distance.

Stage 2:



This is the view immediately after entering the routes.

Step 3:



The front-end page of the application has view and controller units. The developers find react useful to build components which are encapsulated. Encapsulation helps in building the user interfaces which are complex. The parsing of the data through the app is made easier with react. It efficiently helps in rendering the right components when there is a change in the input data.

The Back end is implemented through a framework of flask which is quite famous among the python developers because it is a persistence engine with a fully integrated testing and a powerful asynchronous Web Service client. It provides full support to the MVC model. It can easily scale up any implemented application to further levels. A view object is an object that the application returns, and the user/client can see/use. The object can be of different types depending on the application like Hypertext Markup Language (HTML) / Cascading Style Sheets (CSS). The front end will use the Json to render a view in HTML for the users of the application. A controller interprets user actions and intentions and interacts with the models and views to carry out the action.

1.1 Used tools:

a. Visual Studio Code:

Visual Studio Code is a source code editor developed by Microsoft for Windows, Linux, and Mac operating system. The support that it delivers to handle debugging with version control along with the refactoring of the code helps in handling the visual studio code with ease. It is also customizable, so users can change the editor's theme, keyboard shortcuts, and preferences. Unique features of Visual studio code are:

- Edit, build, and debug with ease.
- Robust and extensible architecture.

b. IntelliJ IDEA:

IntelliJ IDEA is a java integrated environment that helping efficient working. This helps in writing the code with an ability to handle the bugs and deployed exeptions. This is a more user-friendly application to access the shared configurable files. With an impactful testing plan, the probability that we encounter high density of bugs is almost minimal. So, the quality control of the application is optimally maintained.

c. Rest API:

Rest stands for representational state transfer. It follows a Client-Server Architecture. A client server architectural pattern us followed. The rest API uses a HTTP protocol – HyperText Transfer Protocol. Networking applications can be successfully executed. Complex services can be executed through Rest API for transport communication. It can serve as an light alternative for java application

The tasks performed by REST API:

HTTP requests can be made to create the data, further update, read and delete the data. The return values are JSON (JavaScript Object Notation). An appropriate HTTP method is required to perform a correct operation. Few methods to be used:

1. GET – We use this to get the data
2. POST – We use this to create the data
3. PUT- We use it to update the data.
4. DELETE- We use this to delete the data.
5. Paths should also be specified.

HTTP response examples:

200-ok

201-create

204-deleted

400- Bas Request

403-Forbidden

404-Not Found

500-Internal Server error.

d. Flask:

Flask is a python written micro web framework. It is termed as micro framework because it doesn't require any sort of libraries or tool. It doesn't contain any database abstraction layer. The components which provide third party libraries which provides any common functions. It can handle authentications and object-relational mappers.

Flask could be used for the following functionalities:

- There is an integrated support provided for unit testing
- It uses Jinja templating
- It has a compatibility with google app engine.
- With the flask implementation, we can extend the functionality of the application.

e. React JS:

React is a JavaScript library intended for user interface building. It works by creating a virtual DOM in the memory, where all the necessary manipulation takes place. React parses all the made changes and retains only the changes those are actually necessary. Component logic is written in JavaScript instead of templates, so data can be easily passed through the app and keep state out of the DOM. It allows the user to design simple views for each state in the application and React will efficiently update and render just the right components when your data changes. EleNA has map components and it fetches the required route when the input controls changes.

f. Open Street Map REST API:

Open Street Map is a user contributed maps of the world which are free to use. The open street map API calls which are meant to create, update, read and delete. There are primarily three elements which are the basic components of data model of the open street map.

a. Node:
    - It represents a particular point on the surface of the earth which is determined by the latitude and longitude.
    - Stand-alone features can be defined using nodes.
    - They are used to define the shape of the way.

b. Way:
    - A way is a polyline defined by an ordered list of 2 to 2,000 nodes. Linear features like rivers and highways are represented by ways. way
    - The boundaries of areas (solid polygons) like buildings and trees can also be represented by ways.

- The initial and end nodes of the path will be identical in this scenario. We refer to this as a "closed way."

c. Relation:

Describes the relationship between two or more data components (nodes, ways, and/or other relations) in a multipurpose data structure which is a relation.

g. OSMnx library:

One can obtain geographical data from OpenStreetMap and use OSMnx, a Python tool, to project, model, visualize, and analyze actual street networks and other geospatial geometries. With just one line of Python code, you can download and model walkable, driveable, or bikeable metropolitan networks, which you can then quickly evaluate and visualize. Other infrastructure types, amenities/points of interest, building footprints, elevation data, street bearings/orientations, and speed/travel time can all be downloaded and used just as easily.

1.2 Components of view:

Text box: The text box is the area where we enter the source and the destination.

Map View: It gives the route as per the given specifications for the user to view.

Route statistics are displayed as a part of the output.

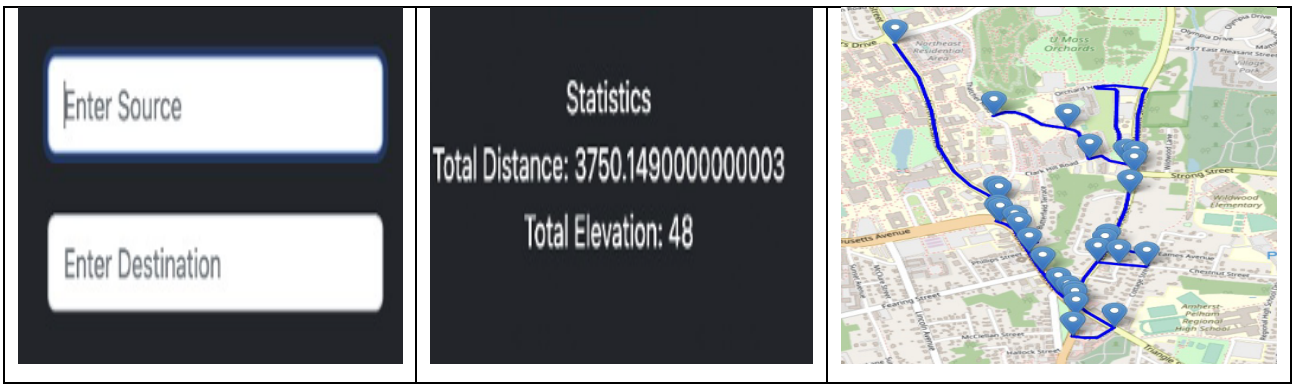


Fig 2. Components of View

1.3 Components of Controller:

The components of Controller in our application includes the following:

Elevation:

It is a min/max toggle button. The elevation is a control button that the user needs to give as an input. Once it is given the maximum or minimum elevation is calculated through the defined routing algorithm in the model and the output is displayed.

Algorithm deployed:

Paths with location coordinates where Node(latitude, longitude)

Path1- Node1, Node2, Node3, Node4, Node5

Path2- Node1, Node2, Node3, Node4, Node5

Obtaining elevations of nodes for each path:

Path1- Elevation(Node1), Elevation(Node2), Elevation(Node3), Elevation(Node4), Elevation(Node5)

Path2- Elevation(Node1), Elevation(Node2), Elevation(Node3), Elevation(Node4), Elevation(Node5)

Now to compute the route with maximum elevation gain:

Elevation gain of Path1 - max_Elevation(Elevation(Node1), Elevation(Node2), Elevation(Node3), Elevation(Node4), Elevation(Node5)) - min_Elevation(Elevation(Node1), Elevation(Node2), Elevation(Node3), Elevation(Node4), Elevation(Node5)

Elevation gain of Path2 - max_Elevation(Elevation(Node1), Elevation(Node2), Elevation(Node3), Elevation(Node4), Elevation(Node5)) - min_Elevation(Elevation(Node1), Elevation(Node2), Elevation(Node3), Elevation(Node4), Elevation(Node5))

Now whichever has the maximum elevation that path will be returned.

If Elevation gain of Path1 > Elevation gain of Path2=> return Path1

Percentage Increase:

It is a slider. It specifies the percentage of the shortest path that they would like to traverse. It's user interactive and it triggers the model.

Submit:

Once all the required information is given, i.e.,, entering the source and destination in the view part of the application, later giving appropriate the elevation gain and the % distance to be traversed of the shortest path, the user hits on the submit button, then through the API's the functions those are defined in the model are triggered.

Fig 3: Controller components.

Model:

In the model, we have implemented Dijkstra's algorithm to obtain the shortest path between starting and ending location. In obtaining the path we computed the latitude and longitude of all the nodes in the path using open Street Map API. To add the elevation and percentage of shortest path as an additional filter to our algorithm we followed open-elevation API to gain elevation of that particular node in the path. We designed algorithm in such a way that we can implement all these filters. For implementing shortest path, we used the percentage of shortest path from the slider. Using that numeric as radius obtained an enclosed area and stored all the paths from that obtain x% of the shortest path for the final one.

On top of this using the minimum and maximum elevation toggle button we mark the final route which has minimum elevation gain for the toggled for minimum elevation and similarly for maximum elevation route too. Firstly, from all these paths we store maximum and minimum elevation node from each path. Further for computing elevation gain we used difference between maximum elevation and minimum elevation. After obtaining elevation gain for all the paths in that radius of shortest path returning the route with maximum elevation gain.

VI. EVALUATION CRITERIA

Evaluation of the application is very important to improve the performance and to understand the credibility of the application. It gives us scope to validate all the functionalities present

which assists us to use the application to the fullest and deliver it such that the objective is met perfectly. The dysfunctional functionalities can be avoided.

6.1 Functional Requirements evaluation:

In the requirement specifications, we made sure that all the functionalities defined are met.

i. The user interface is interactive as the user is able to give a valid start and end locations and submit the route request.
ii. The elevation gain is successfully toggled between maximum or minimum as per the user requirements and the change in the route is observed
iii. A slider which denotes x% of the required path to be taken is provided in the Front-end and works as expected. The slider guides smoothly.
iv. An algorithm is implemented such that it computes the route based on the given inputs and the respective route statistics (elevation gain and total distance) are displayed successfully.
v. A map that shows the route between the start location and end location as per the required inputs with markers
vi. The map is magnifiable and movable.

6.2 Non-functional Requirement evaluation:

i. Readability:

The entire code is very well defined. We see the naming convention is followed in the code i.e., proper names are chosen for representing the variables, functions, and other entities in the code. We see that the naming of the functions is generic, and distinguishable from each other and improves the appearance making it visually appealing by using small names by avoiding abbreviations. The presence of comments in the source code helps in easier understanding with less effort.

ii. Usability:

Usability refers to the ease by which a user interacts with a product. It is much important to initially understand the objective with which an user interacts and make sure the whole process is smooth. We make sure that the application is more user-friendly. The usability

for the application developed is improved on iterative basis. With the UI we have, we ensured that it's appealing to use and easier to navigate between the tags.

iii. Version control:

Version control is very much essential for the tracking of changes that we perform. It ensures that all the collaborators of the project are working on the right version of the file. It helps in accelerating of the product delivery.

iv. Modularity:

Modularity ensures that systems are built in a cohesive manner with loosely coupled components. The components work together with a proper interaction and there are separate modules generated for each function.

Various main components associated with the code are:

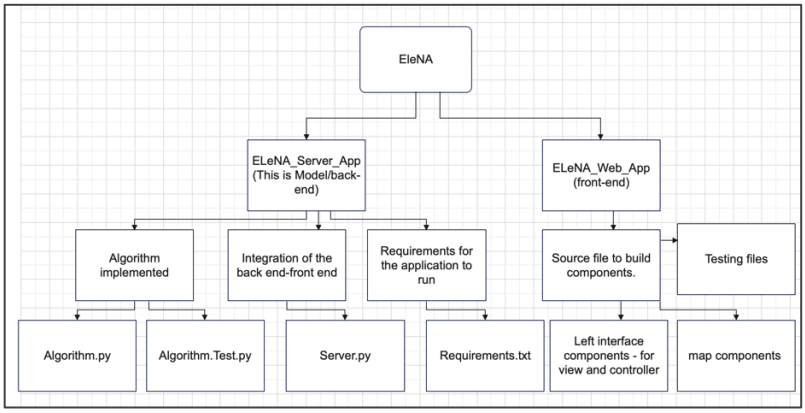


Fig 4: Modularity of the components

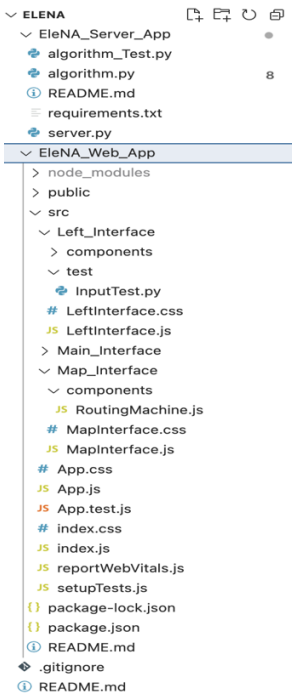


Fig 4: Modularity of the components

v. Performance:

The performance of the application refers to how quickly it renders the path once the inputs are given, and the model is triggered. It is observed for various samples and the time stamps are calculated.

S.No	Source	Destination	Elevation	% of shortest path	Time taken to generate the route
1	Worcester Dining Amherst , MA	Franklin Commons, Amherst, MA	Min	75%	1.75 sec
2	Brandywine Apartments, Amherst, MA	Boulder Apartments, Amherst, MA	Max	60%	2.05 sec
3	University of Massachusetts, Amherst, MA	Blue Wall, Amherst, MA	Max	100%	2.05 sec

4	Hadley Rd, Amherst, MA	Brittany Mnr drive, Amherst, MA	Max	75%	1.05secs
5	Subway, Amherst, MA	Morrill Science Center, Amherst, MA	Min	50%	2.25 secs

vi. Debuggability and Testability:

It is the process in which we identify the failure of implemented code. It is a composition of validation and verification of software and Debugging is the process that we fix the bugs those are identified during testing. Proper MVC architecture, readability and improved modularity amplifies the ability to test and debug comfortably.

There are various tests conducted:

- Alpha testing
- Beta Testing
- Selenium Testing
- Unit Testing

Alpha testing: The functionalities of the application and the proper working of the controllers are ensured during the execution of the web application.

Various inputs are given and the responses are observed, while the changes are made iteratively if there is any unsatisfactory outcomes.

Validations triggered for certain corner cases:

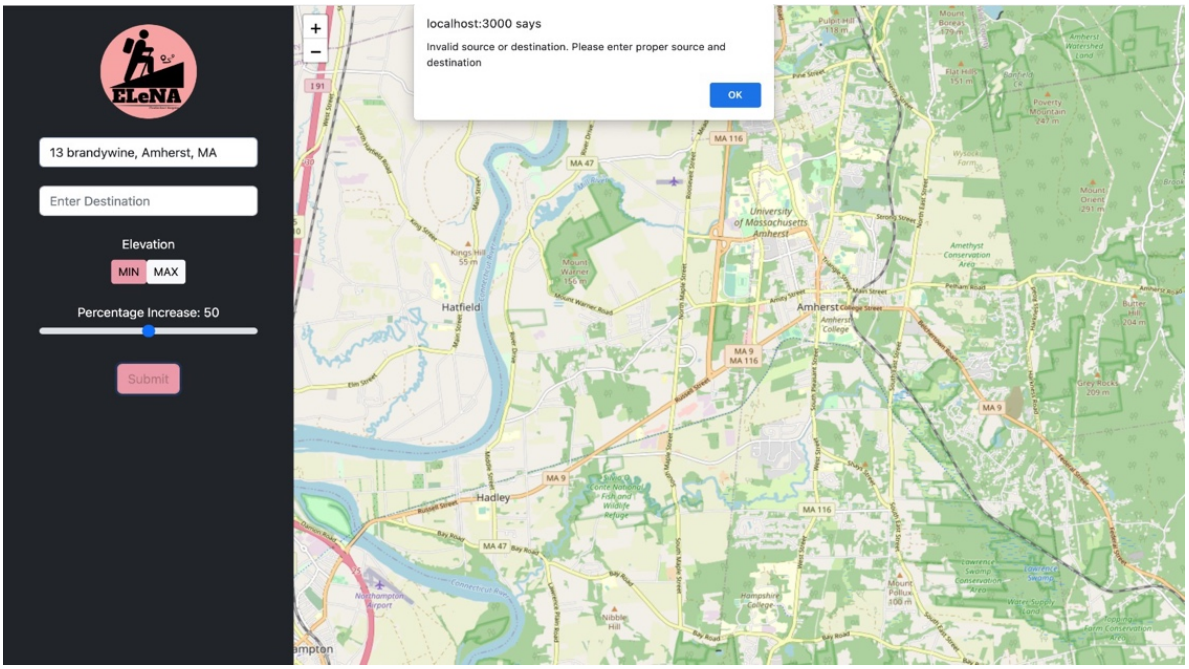


Fig 5: Destination is empty

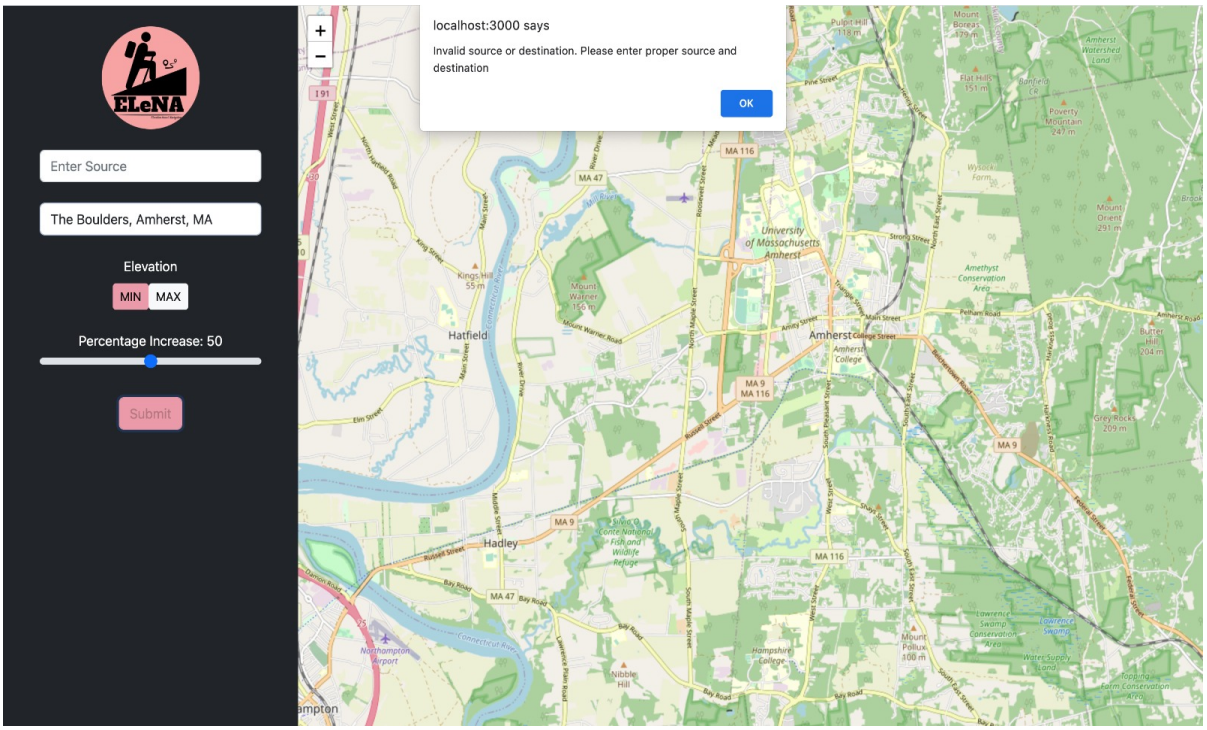


Fig 6: Source is empty

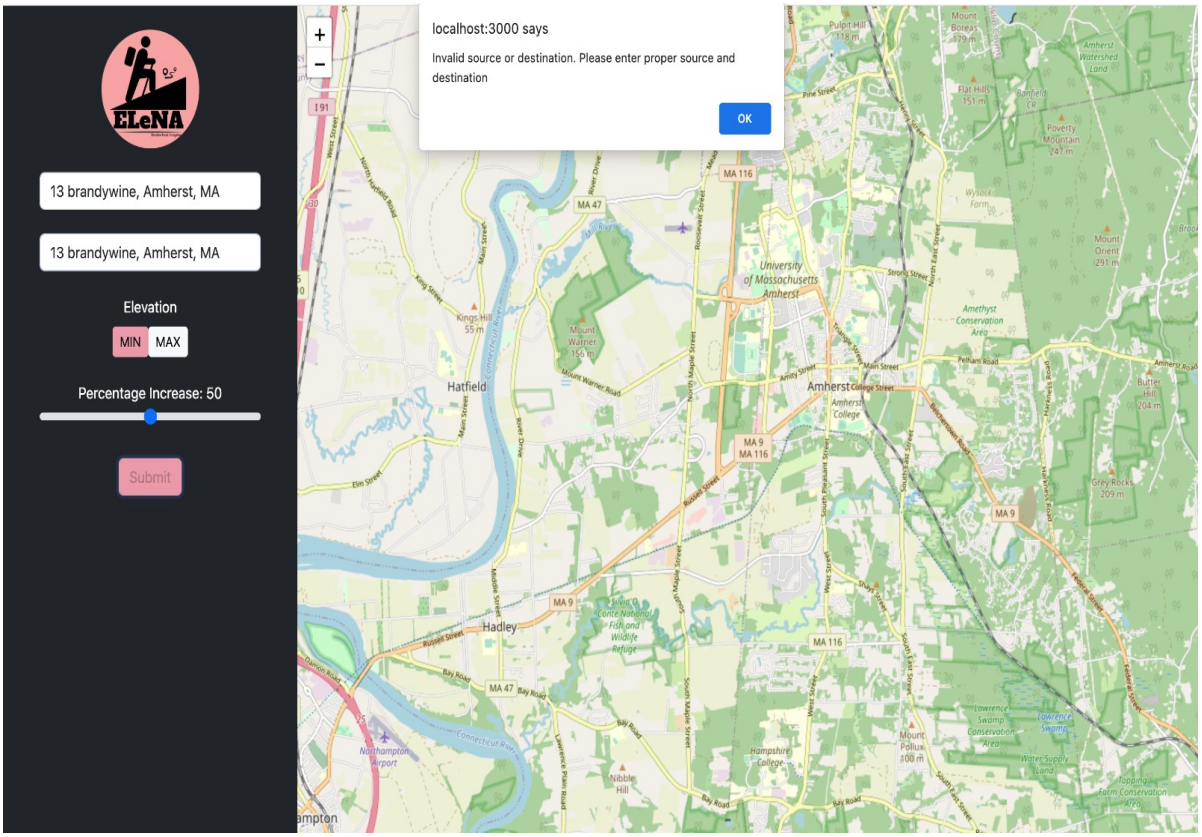


Fig 7: Same source and destination is empty

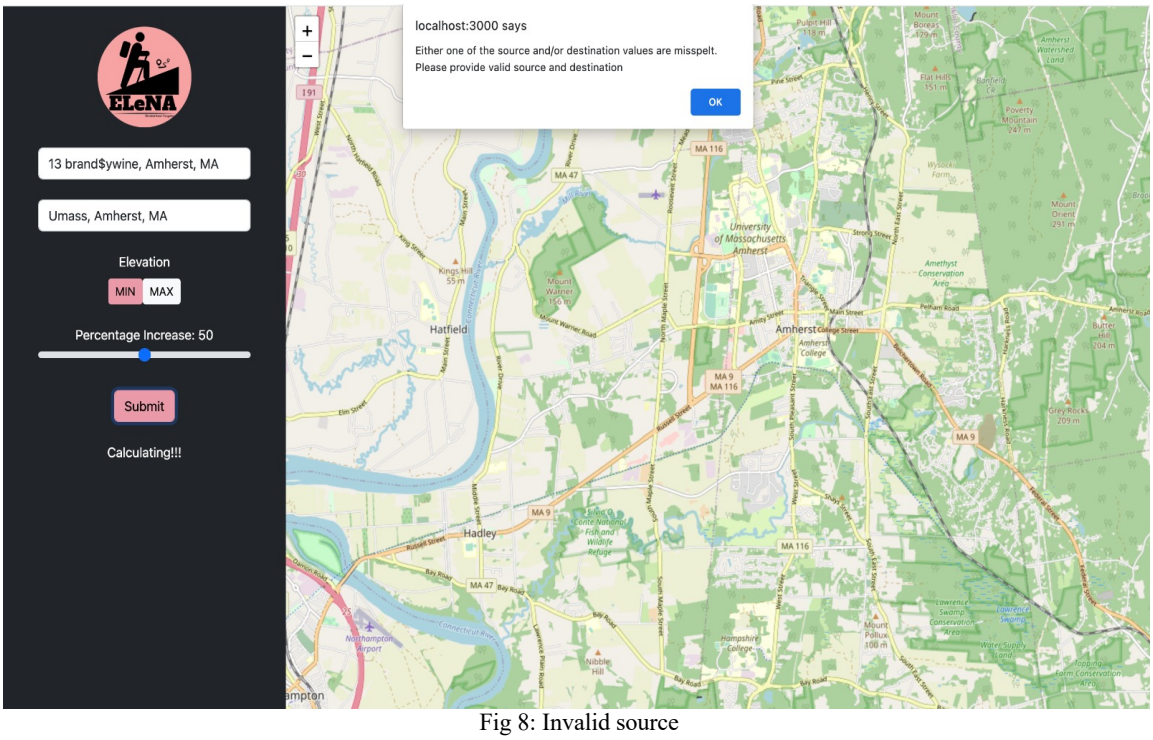


Fig 8: Invalid source

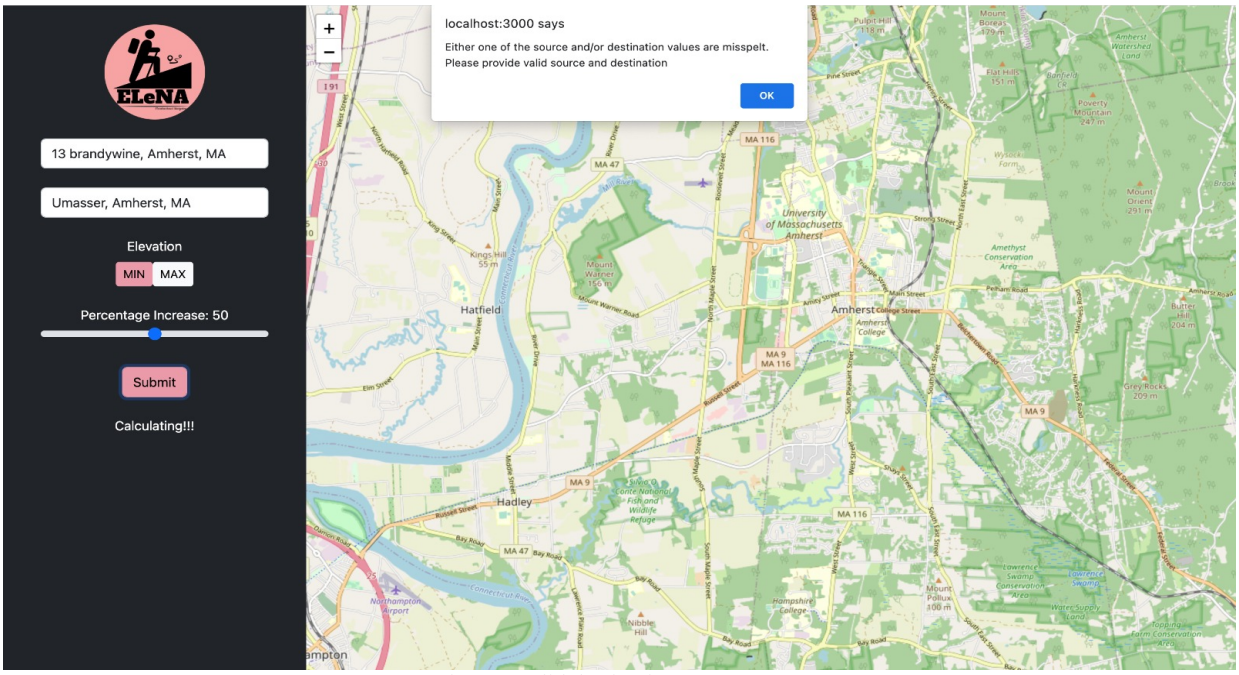


Fig 9: Invalid destination

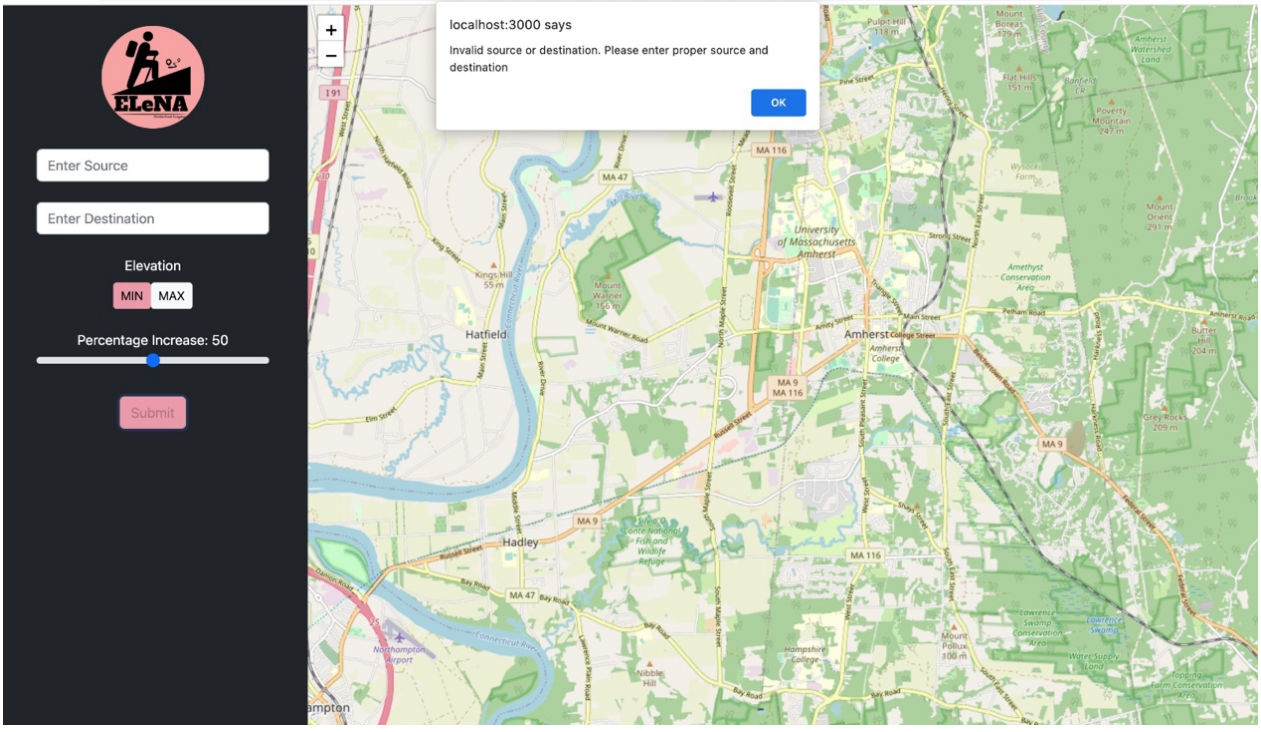


Fig 10: If no information of source and destination is given

Selenium Testing:

We have verified the below scenarios both manual and using the selenium test driver:

Selenium Testing:

- Testing Happy path(end-to-end scenario with valid location) when max elevation is provided
- Testing ALERT popup when source and destination are provided the same
- Testing ALERT popup when source and destination are provided black
- Testing ALERT popup when the source is spelled incorrectly
- Testing ALERT popup when the destination is spelled incorrectly.

```
(ox) priyamarla@Priyas-MacBook-Air test % python3 InputTest.py
/Users/priyamarla/Documents/fall2022/520/Project/EleNA/EleNA_Web_App/src/Left_Interface/test/InputTest.py:8: DeprecationWarning: executable_path has been deprecated, please
 pass in a Service object
  driver = webdriver.Chrome("/Users/priyamarla/Downloads/chromedriver.exe")
Obtained distance value is correct for minimum elevation gain
Obtained elevation value is correct for minimum elevation gain
----Reached end of testing when minimum elevation gain is selected----
Obtained distance value is correct for maximum elevation gain
Obtained elevation value is correct for maximum elevation gain
----Reached end of testing when maximum elevation gain is selected----
Alert exists in page when source and destination are blank
----Reached end of testing when source and destination are provided blank----
Alert exists in page when source and destination are same
----Reached end of testing when source and destination are provided same----
Alert exists in page when source is spelled incorrectly
----Reached end of testing when source is spelled incorrectly----
Alert exists in page when destination is spelled incorrectly
----Reached end of testing when destination is spelled incorrectly----
(ox) priyamarla@Priyas-MacBook-Air test %
```

Unit testing:

Unit Test for the algorithm:

We have covered the following test scenarios as part of our unit testing:

- Checking if correct latitudes and longitudes are being generated for start and end location.
- Checking if minimum elevation is generating the proper route.
- Checking if maximum elevation is generating the proper route.
- Checking if x percent of the distance is being generated correctly.
- Verifying if the final route nodes are correct.

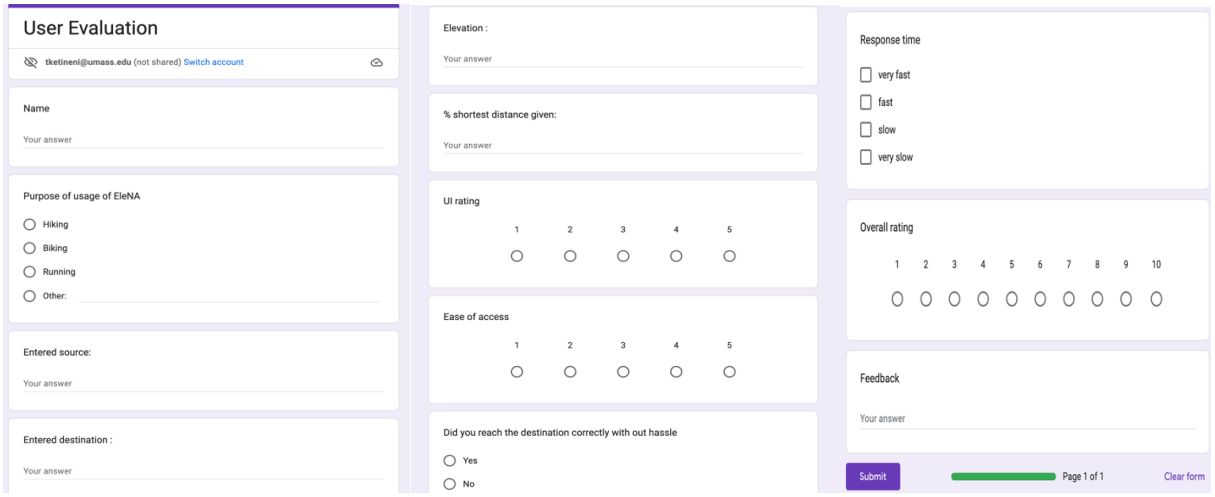
```
(base) yoshitavarma@yoshitas-MacBook-Air EleNA_Server_App % python algorithm_Test.py
/Users/yoshitavarma/opt/anaconda3/lib/python3.8/site-packages/scipy/__init__.py:138: UserWarning: A NumPy version >=1.16.5 and <1.23.0
 is required for this version of SciPy (detected version 1.23.5)
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion} is required for this version of "
.....
----------------------------------------------------------------------
Ran 5 tests in 54.431s

OK
(base) yoshitavarma@yoshitas-MacBook-Air EleNA_Server_App %
```

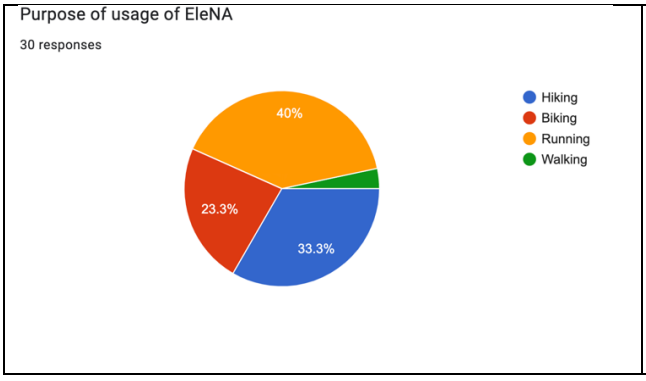
Beta testing:

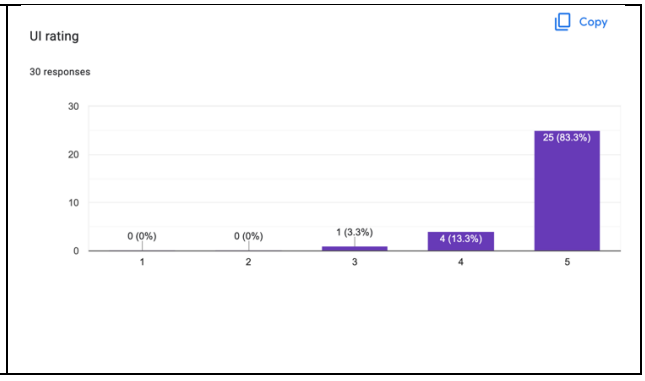
Our application is tested at multiple stages and the feedback is obtained through a questionnaire. We have constructively used their response to improve the performance of the application.

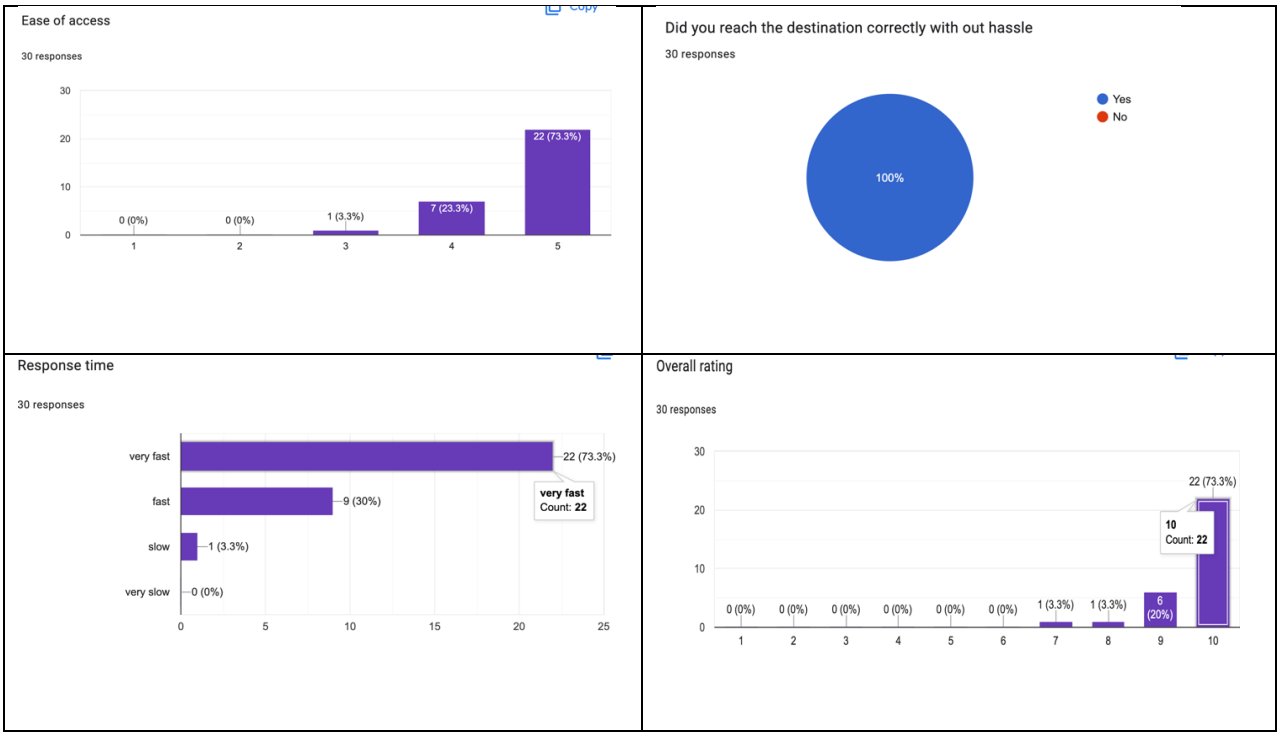
Here is the questionnaire:



Beta Testers Review Statistics:





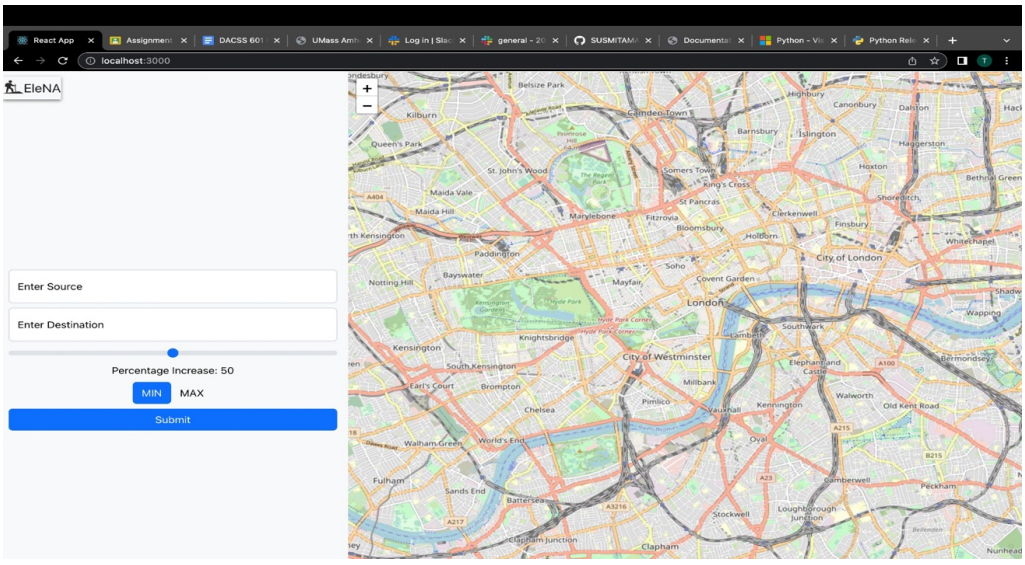


From the statistics it can be observed that most users who used EleNA are runners and hikers. More than 80% of the testers liked the user interface and gave a 10/10 , 70% of them found the application's responsive time to be quick.

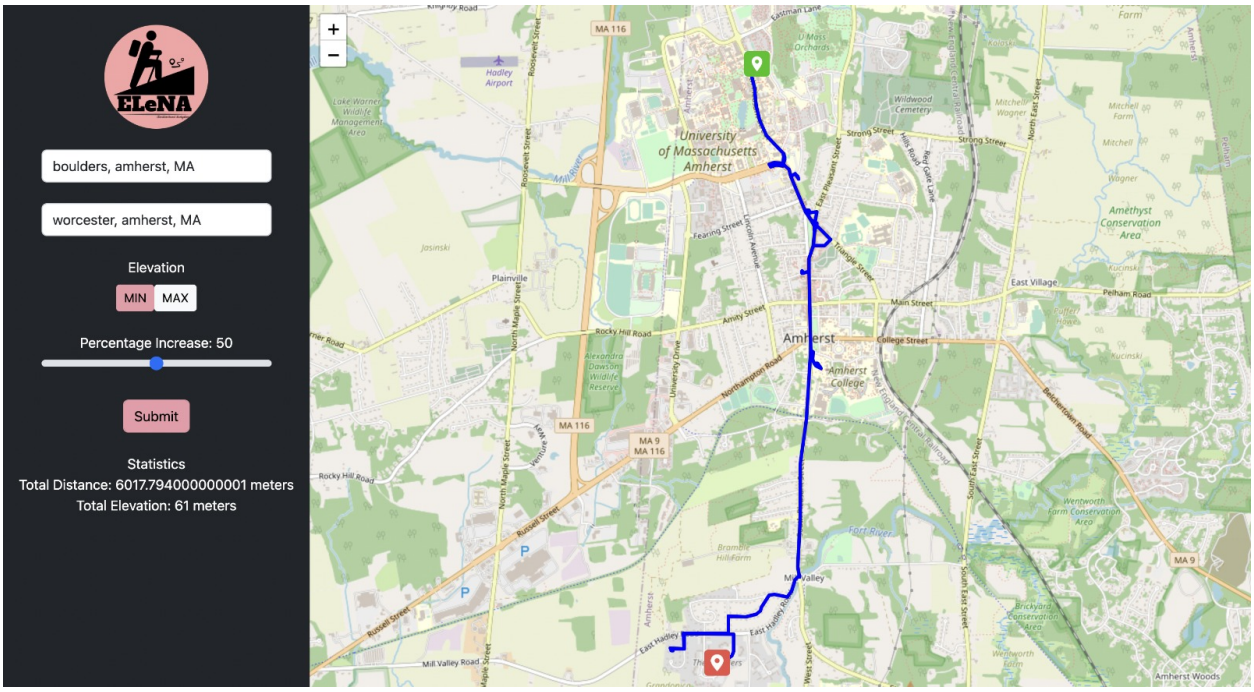
Summary:

We have made many alterations from the beginning based on the feedbacks and improved our application.

The initial UI was picture seen below, this is our first implementation and the first version that we have released for user testing.



After taking feedback on iterative terms the final updated file is given below



Feedbacks from users:

It was as helpful as All trials app in some areas
Perfect
Good stuff!
First class!
The application felt detail oriented, it would be great if you can remove so many markers
The application is so interactive and interesting as it provides advanced features like elevation. Which is very useful for advanced athletics and other purposes. All the best to the team and Keep up the good work.
Was helpful to do an extensive cardio before hitting gym at the campus.
the UI can be much more attractive and it can be much better
Useful web application

A simple yet innovative web application useful for bikers, hikers and runners
amazing
App provided a route with required elevation and distance
Really useful for traveling in the campus in trials.
Very helpful to travel in the nights in the trials
recommended
It was as helpful as All trials app in some areas
Perfect
Good stuff!

VII. CONCLUSION

EleNA - Elevation-based navigation system is of great use to people who would like to travel from one place to other. While other websites give us the proper route to travel, Elena has an added advantage of the elevation factor which calculates the routes based on the user preference of it. Some people might want to have a good hike with maximum elevation and go on a long route instead of the short routes, whereas some would like to use the shortest path with no elevation. All these user requirements can be satisfied with the use of EleNA.

-Susmita Madineni
-Tejaswini Ketineni
-Veera Venkata Naga Padma Priya Marla
-Yoshita Varma Annam
(The Girl Power)